Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 32 (2014) 1050 – 1055

Procedia Computer Science

2nd International Workshop on Survivable and Robust Optical Networks (IWSRON)

# A Comparison of Software Defined Network (SDN) Implementation Strategies

Muhammad H. Raza[a,*], Shyamala C. Sivakumar[b], Ali Nafarieh[a], Bill Robertson[a]

[a]*Internetworking Program, Faculty of Engineering, Dalhousie University, Halifax, B3H 2Z9, Canada*

[b]*Department of Finance, Information Systems, and Management Science ,Saint Mary's University , Halifax, B3H 3C3 , Canada*

**Abstract**

Software defined networking (SDN) is an emerging approach to handle data forwarding and control separately. The notion of programmability has central importance in SDN. Two implementation strategies; proprietary and open source, are shaping the trends of the adoptability of SDN by major hardware manufacturers. A group of leading vendors believes that loose coupling between the logical and physical layers of a network hinders the proper provision of physical resources and suggests a proprietary fix to this problem. The other group regards the notion of openness as s key feature of SDN. This paper compares and contrasts these two implementation strategies of SDN by identifying their respective operating principles, features of the product lines, and weakness and strengths.



Selection and Peer-review under responsibility of the Program Chairs.

*Keywords:* Software Defined Network; Open Source; Proprietary; Overlay; Logical; Physical

## 1. Introduction

SDN has emerged as a technology trend that has attracted service providers, researchers, hardware manufacturers, software developers, and users with an unseen precedence. Traditionally, computer networks are managed with the art of dealing with complexities by adding more protocols to protocol suites to handle complications in network operation. SDN produced a lot of excitement in the networking community as it introduced the element of modularity in networking that never existed before. This results in the replacement of a bundle of mingled up protocols and system components to reusable abstractions.

In conventional networks, network devices deal with both data transfer and control functions. The changes in network infrastructure such as large-scale addition of end systems, and real and virtual networks are difficult to handle in conventional communication networks. SDN is known for separating the data and control functions of networking devices, such as routers and switches by interacting through Application Programming Interface (API) between the data and control functions as explained in Figure 1.

It is important to analyze the defining characteristics of SND. A common logical architecture at SDN devices and

* Muhammad H. Raza. Tel.: +1-902-494-6869.
*E-mail address:* hraza@dal.ca


Selection and Peer-review under responsibility of the Program Chairs.
doi:10.1016/j.procs.2014.05.532

a protocol between an SDN controller and the network devices are the two key aspects of SDN[2] as shown in Figure 1. Meaning that all switches, routers, and other network devices to be managed by an SDN controller should have a common logical architecture. The unifying feature of SDN enables different vendors to implement this logical architecture in different ways to build network devices to operate under an SDN controller that sees a uniform logical switch function. For having overall picture of the network, a central controller in SDN is a better way to handle localized problems in networks such as congestion or spectral noise. Due to the autonomous nature of switches, switches may make routing decisions as per their limited view of the network and such decision may make sense to switches only. When such decisions are viewed from a larger perspective of the entire network, such decisions may prove bad choices. An SDN controller, being aware of the overall situation of a network, can assist in providing alternative routing options.

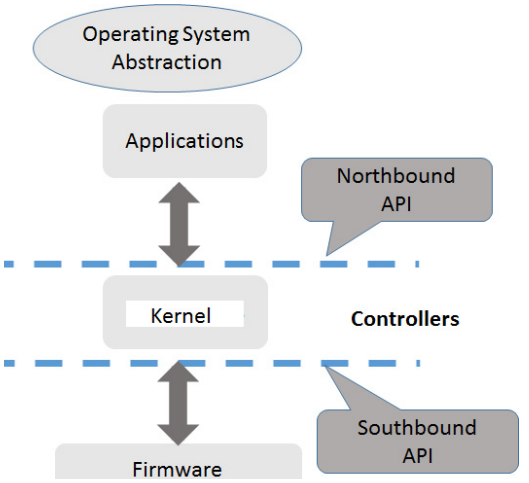


Fig. 1. Schematic components of SDN

A few technological factors, such as visualization and an increase in the number of mobile devices, have been behind the technological push of SDN. Visualization has revolutionized the handling of traffic flows as compared to the handling of flows by traditional client-server setup. The change in the location and intensity of flows over time requires a flexible approach for successful network resource management. The number of handheld devices like smart phones, tablets, and notebooks has increase many folds to put pressure on enterprise resources. Network resources change rapidly and management of Quality of Service (QoS) security becomes challenging.

**Effects of SDN on Businesses, Service Providers, and Customers:**

From business perspective, SDN provides benefits of lower operating expense and capital expenditure, but these business benefits are not without risks. Thats why customers are hesitant to deploy SDN-based technologies in their networks because of the risks impacting production traffic. Proof of this hesitation by customers is available in the survey report, 2013 SDN Survey: Growing Pain, which shows 33 percent of customer has no plan to test SDN in near future and 47 percent find product immunity as a barrier in adopting SDN[5].

Service providers are interested in SDN, because resource-intensive applications are causing network traffic to grow exponentially and this increases the demand of resources on existing network. The dynamic allocate of network resources to higher-priority applications has its own challenges in addition to the challenge of differentiating between critical and noncritical applications.

This situation puts customers under pressure to look for solutions to make networks applications-aware by intelligently monitoring and routing the network traffic. The role of SDN becomes prominent as it makes the network programmable, dynamic, modular, abstraction-based, and application aware. The adoption of SDN comes to customers at a cost, as customers have not embraced the technology in production environments yet, therefore many fears and hurdles exist in accepting SDN technology. A centralized approach simplifies the management of complex

flows and enables programmability at the cost of various drawbacks such as the requirement of major changes in the production network, integration of networking, programming skills set and support related problems from multi-vendor situation.

One of the options in fulfilling the requirement of a standard protocol between the SDN controller and network devices is OpenFlow[1]. In fact, OpenFlow is both a protocol between SDN controllers and network devices, as well as a specification of the logical structure of the network switch functions. As a matter of fact, SDN does not depend solely on OpenFlow, and it will still make a network programmable. But it may not affect the underlying networking hardware in the same way OpenFlow can without extra arrangement. OpenFlow as part of SDN may be instrumental to commoditize the switches and routers. And it had a big impact on the networking vendors such as Cisco, HP, IBM, Arista and Juniper[1].

A divide exists in the networking industry over the implementation strategy of SDN. One camp supports the openness at the cost of isolated overlay network from the physical network at the bottom and the other camp emphasizes on an interaction with the physical network at the cost of restrictions on openness. This paper presents a comparison among various SDN strategies by highlighting the differentiating factors, operating principles, and market placement.

The rest of the paper is organized as follows. Section 2 analysis the proprietary SDN implementation. Section 3 investigates SDN strategy hat is based the open source and non-proprietary concepts. Section 4 presents discussion and comparison of the two SDN strategies. Section 5 concludes this paper.

## 2. Proprietary SDN Strategy

One of the two distinctive SDN tends in networking world with respect to SDN is to have products with proprietary software components. This approach gives importance to programmability but puts restrictions to the openness by having propriety components in a programmable infrastructure. The customers are interested in the programmability of computer networks while the industry is deciding about the placement of OpenFlow and SDN in the current network system. Therefore, while OpenFlow and SDN are important developments, the real thing that will get customers excited is exposing the intelligence that's already there in the network and be able to program networks as per their needs. Having such an approach means that instead of dealing with protocol configuration such as border gateway protocol (BGP) and virtual local area networks (VLAN) setups, a user just passes on a network the requirement of a connection between two points under certain service level agreement (SLA). The underlying network will make the arrangement to complete this networking task. All industrial players in the networking world agree on the requirement of programmability, but they differ in how this programmability should practically be implemented. Some hardware vendors want to use proprietary components in implementing programmability but the other type of vendors may consider it a compromise on being open source[3].

Many vendors, such as VMware and Cisco, sell proprietary SDN controllers along with higher-level software applications as a part of their programmable networking system. Cisco offers, in particular, a range of products to suite to various levels of networking as described in the following paragraphs.

Cisco provides a hybrid approach to SDN, in which the traditional control plane continues to exist and an external controller enables programmability and application flow management for specific business requirements. As Cisco has indicated acceptance to open source SDN community by embracing OpenFlow, but Cisco is also exploring proprietary systems within the context of programmability. It is clear that the OpenFlow is just one component of SDN[5]. Cisco wants to combine both technologies to build various products to create an intelligent network that delivers an efficient, scalable and highly available environment. Cisco offers One Platform Kit (OnePK), which is a software developer's kit that provides a consistent set of application programming interfaces (APIs) exposed on all of the Cisco's operating systems, IOS, IOS-XR and NX-OS. With OnePK, network management applications are able to do meaningful management at least in an all Cisco environment. Earlier, a lack of good API access has left network management vendors to use poorly documented simple network protocol management (SNMP) interfaces that varied across products. As a result, network management products can do a lot more than acting merely as network monitoring applications.

The Cisco Open Network Environment (ONE) is a customizable framework that offers programmability, and abstraction at multiple layers. The Cisco ONE offers a choice of protocols, industry standards, use-case-based deployment models, and integration of functions. The foundation of Cisco ONE is policy programming for a dynamic feedback loop of user, session, or application analytic. Figure 2 explains the Ciscos' feedback-based SDN strategy

and differentiates it from the limited approach of just separating control and data plane. Thats why, a recent InformationWeek survey of 250 IT professionals showed that Cisco Systems SDN vendor strategy ranks as the number one SDN strategy[5].

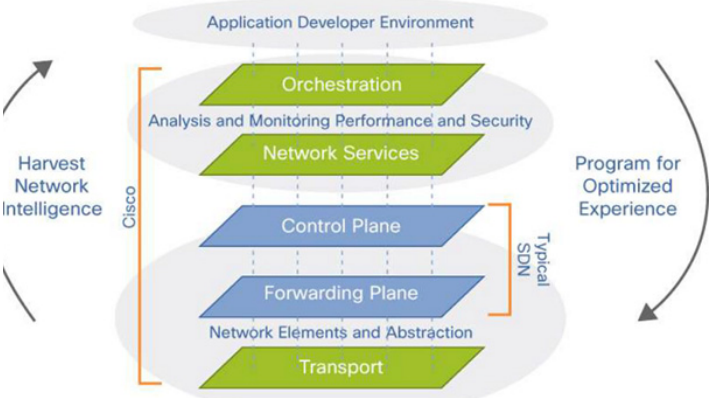


Fig. 2. Difference between Cisco and non-Cisco SDN strategies[4]

The Cisco ONE is delivered through a variety of mechanisms, including APIs, agents, and controllers. The Cisco approach complements traditional approaches to software defined networking (approaches that primarily focus on decoupling the control and data planes), while also encompassing the entire solution stack from transport to automation and orchestration.

OpenStack enables servers, networking, and storage systems work together in a private cloud environment. Thats why, in addition to supporting OpenFlow, Cisco also supports the open source entity of OpenStack in cloud computing by crating APIs to their Nexus switches and building extensions to OPenStack. This conviction appears a strong driver behind Cisco's joint venture, VCE, with companies like EMC and VMware where OpenStack has been used. Cisco has continually added features like vPath to the 1000V, which can be used to add Layer 4 to Layer 7 services like load balancing and firewalls its virtual switch to provide a step towards programmability. Cisco is also supporting Virtual Extensible Local Area Network (VXLAN) to enables overlay networks.

After presenting an analysis of the proprietary SDN approach, the non-proprietary SDN approach is analysed in the next section.

## 3. Open Source and Non-proprietary

The followers of the open source and non-proprietary SDN believe that the main purpose of SDN survives by being available to all as an open source provision with no hidden proprietary components so that the independent use and development of SDN could grow further.

Some hardware vendors might like the notion of being open source or not, but the competition in the market is forcing them to consider open source options as their competitors have already found a fit of the open standard in their product lines. Based on this aspect, NSX product of VMware is capable of creating a virtual network overlay that is loosely coupled to the physical network underneath. Similarly, OpenContrail by Juniper is an SDN controller freely available through an open-source license. A re-branded version comes with services and support on per socket cost basis.

The open programmable SDN product suite by Big Switch enables easy adaptation of new network applications in an easier way as compared to the adaptation of new applications with traditional and non-programmable networks. This hardware platform-independent suite supports open standards and APIs including OpenFlow. HP also backs up open standard and offers an OpenFlow-enabled SDN controller and switches.

The Big Switch Networks product suite includes Big Network Controller (BNC), Big Tap, and Big Virtual Switch (BVS). BNC scales to more than a thousand switches and 250,000 new host connections per second. Big Tap is a unified network monitoring application designed to provide enterprise-wide network visibility. BVS is a data center network virtualization application designed for automated network provisioning. Big Switch makes use of both popular open sources; OpenFlow and Floodligth to provide network abstraction for the physical infrastructure, policy-based functions across the fabric, and centralized intelligence for programmable networks. To accomplish these features, Big Network Controller includes an OpenFlow southbound programming interface, a RESTful API for northbound communications, and is based on the Floodlight open source controller code available under the Apache 2.0 license.

The next section presents a discussion on these two leading SDN trends (proprietary and non-proprietary) and identifies key differences and strengths of these two trends.

## 4. Discussion on Comparison

Some industry players believe that many of the benefits of SDN can exist without using OpenFlow. Products from Nicira, Juniper, Cisco and many other SDN startups do not depend on the lowest level of a network and offer programmability for scalable and virtualized infrastructure without OpenFlow. Such products have features that are easier to implement for enterprises and cloud customers. This approach is convenient for those businesses that have the resources to program and support entirely new networking code for new routers built on commodity hardware. This approach is also attractive for those companies that do not want to replace their existing equipment base to buy a new OpenFlow-based network product.

The followers of proprietary concept criticize the approach of creating a virtual network overlay (NSX product of VMware) that is loosely coupled to the physical network underneath for poor scalability. The question of software switches and virtual network overlays being enough to handle high-performance environments really depends on the situation rather than the generic scalability capacity of a networking product.

The favorers of open source and non-proprietary school of thought criticize the vendors who persist to keep proprietary components in their product. For example, Cisco is criticized for its business stakes in the following manner. The core routers sit at the heart of the data center and have the main control over information distribution . The data moves from the core of a network to the end of a row (EOR) of racks in a data center, then on to the switches at the top of each rack (TOR). The core is the most profitable business for Cisco. Cisco has three different operating systems (IOS, IOS-XR and NX-OS) that are all differing technologies. These OSs have different release cycle cadences. Any programmable SDN requires to comprehend all of these OSs to tie the release cycles or consistent updating of OSs. This might need a consistent effort to convince those customers who relish stability and longevity in releases. The fear of some SDN products being over featured is also there where the TOR or EOR switches no longer need the deep set of features.

In spite of this criticism, Cisco is in a good position to leverage the SDN opportunity. Cisco can do so by aligning its forces to pro-actively help the industry understand the required changes and using its organized human resource and technical channels to facilitate availing the SDN opportunity. Cisco has an opportunity to take over the competition with an SDN-based TOR switch. Todays networking products have been focused on converged infrastructures and encapsulating Fibre Channel into Ethernet packets (FCoE) in order to reduce complexity. A shift in Ciscos focus from converged environments to SDN and a lower cost of SDN TOR can prevent some of the potential defection of customers. Just adding SDN to its current line of switches line will not present a compelling opportunity for customers and will open the door to other vendors to take their space.

With Dynamic Fabric Automation (DFA), Cisco is the only vendor in the market with a strategy to orchestrate physical tunnelling functions in network hardware with software network agents such as the Nexus 1000V. This allows the deployment of overlay networks that connect both virtualized platforms such as OpenStack or VMware to non-virtualized devices and servers. Instead of supporting virtual workloads in a cloud platform like vCloud or OpenStack, Cisco can support any workload, anywhere. DFA looks to be a strong product that certainly meets customer needs, goes beyond competitive products and plays to Cisco's strength of integrating the physical and virtual networks. Unfortunately, the choice of a non-standard and proprietary encapsulation is criticised as a significant drawback.

Though SDN has been No. 1 on the list of technologies that will create the next generation of data center networks, the whole approach may appear flawed because SDN systems are based on abstractions of existing models of the

network. This attribute limit the ability of SDN to merge management of physical and virtual network assets. The focus of SDN on the whole network rather than managing one network box at a time is another challenge that limits the programmability and ability to apply management policies to the entire network from a single point.

A summary of the key features of the comparison between these SDN strategies is given in Table 1, where a comparison is presented from the point of views of network control, feedback from physical layer to logical layer, stability and vendor support, and the situation of standardization.

Table 1. A comparison of proprietary and non-proprietary SDN strategies.

| SDN Strategy | Control) | Feedback | Support | Standardization |
|---|---|---|---|---|
| Proprietary | Restricted | Yes | Yes | Under control of one vendor |
| Non-proprietary | Open to all | No | Limited without payment | Wait and see |

## 5. Conclusions

This paper compared implementation strategies of SDN by identifying their respective operating principles, features of the product lines, and weakness and strengths. No-proprietary implementation comes at a cost but provides a stable and backed up by support products. The open source products speed up the implementation but a lack of feedback between logical and physical layer cannot be ignored. Hybrid approach has potential to lead the trend.

## References

1. Nick McKeown, Software-defined networking. *NFOCOM keynote talk* 2009.
2. https://www.baycollege.edu/Academics/Areas-of-Study/Computer- Network-Systems/Faculty/Linderoth/2013-sdn-survey-growingpains. aspx, lastly accessed on Mach 28, 2014.
3. Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review* 2008;**38(2)**:6974.
4. Bob Lantz, Brandon Heller, and Nick McKeown. A network in a laptop: rapid prototyping for software-defined networks. *In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, ACM* 2010.
5. http://www.cisco.com/web/CA/solutions/trends, lastly accessed on Mach 28, 2014.